3-22-2018

Securing Data in Transit using Two Channel Communication

Clark L. Wolfe

SECURING DATA IN TRANSIT USING TWO CHANNEL COMMUNICATION

THESIS

Clark L. Wolfe, Capt, USAF

AFIT-ENG-MS-18-M-069

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-18-M-069

SECURING DATA IN TRANSIT USING TWO CHANNEL COMMUNICATION

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Clark L. Wolfe, B.S.E.E.

Capt, USAF

March 2018

AFIT-ENG-MS-18-M-069

SECURING DATA IN TRANSIT USING TWO CHANNEL COMMUNICATION

THESIS

Clark L. Wolfe, B.S.E.E.
Capt, USAF

Committee Membership:

Dr. S. Graham
Chair

Dr. R. Mills
Member

Dr. S. Nykl
Member

Abstract

Securing data in transit is critically important to the Department of Defense in todays contested environments. While encryption is often the preferred method to provide security, there exist applications for which encryption is too resource intensive, not cost-effective or simply not available. In this thesis, a two-channel communication system is proposed in which the message being sent can be intelligently and dynamically split over two or more channels to provide a measure of data security either when encryption is not available, or perhaps in addition to encryption. This data spiting technique employs multiple wireless channels operating at the physical layer, allowing traditional layers above to run seamlessly over it.

Eight data splitting policies are developed with preliminary evaluation of their effectiveness in combating three common cyber security threat scenarios to include eavesdropping, jamming and man-in-the-middle attacks. These policies are then implemented in a simple proof-of-concept communication system simulation. Moreover, a framework is proposed for measuring and classifying the level of integrity, confidentiality and availability that is provided by each policy. While additional discussions present and evaluate potential packet structure, more possibilities for dynamic tunability of the developed policies and any potential vulnerabilities introduced by these data splitting schemes. Lastly, a simulation test-bed is constructed to allow for implementation and testing of future policies. These data splitting techniques could provide additional options to increase data-in-transit security for unencrypted systems operating in contested environments.

Table of Contents

Page

List of Figures

List of Tables

SECURING DATA IN TRANSIT USING TWO CHANNEL COMMUNICATION

I. Introduction

1.1 Introduction

The department of defense requires reliable and trusted communications capabilities. Everything from battlefield radio traffic to electronic mail must be secured using modern day network security technology. In 2012, Secretary of Defense, Leon E. Panetta, stated that Modern armed forces cannot conduct high-tempo, effective operations without reliable information and communication networks and assured access to space and cyberspace. [15]

To this end, many secure communication technologies have been developed for use in various mediums. Three of the most common communication security threats that must be protected against are interception (eavesdropping), injection (man-in-the middle) attacks and jamming attacks, also known as Denial of Service (DoS). The most common solution to interception involves encryption of the data. However, encryption alone does not overcome jamming. This paper presents a novel concept which utilizes multiple channels to provide a secure and more reliable communications path without the need for employing modern day encryption protocols.

1.2 Motivation

There are many situations in which military and commercial networks require a level of secure communications, but are unwilling or unable to provide for the additional bandwidth and processing power required for a robust industry standard

encryption. In these spaces, there exists the potential for multi-channel communication to provide a level of security that could be sufficient without the need for encryption.

One such example of a military need exists in the operation of unmanned aerial vehicles (UAVs), or drones. These UAVs, such as the MQ-1 Predator, provide video surveillance feeds of the battlefield to combatant commanders for hours at a time. That video must be transmitted from the drone which is often flying above unfriendly or contested terrain which provides adversaries the opportunity to monitor or interfere with these communications. In fact, in 2009, it was discovered that insurgents in Iraq were utilizing commercial off-the-shelf hardware and software to intercept these live video feeds from U.S. predator drones. These communication channels did not employ encryption at the time as encryption was found to be prohibitively expensive due to the additional hardware and bandwidth that was required. [6]

Another area of interest for the use of multi-channel communication is in remote sensors networks that are often used in the commercial world to monitor or control small low-power nodes. These sensors, or nodes, are often too power limited, bandwidth constrained or rudimentary to justify the power and latency overhead of robust encryption and decryption operations. Rather than employ the dedicated hardware or power required for encryption, operators may simply leave them open to attacks such as those described above. [14]

1.3 Problem Statement

The threat against traditional communication systems, combined with the overhead of encryption, presents an opportunity for new and innovative electronic messaging schemes in which a measure of security can be provided in conjunction with, or as a substitute for encrypted communication. There are currently no known multi-channel

communication schemes that provide security through multi-channel techniques to securely transmit data.

1.4 Approach

A research approach to multi-channel communications should answer some fundamental questions regarding the design and operation of such a system. The questions that this research seeks to answer are as follows (Note: the number of channels has been set at two in order to simplify analysis and presentation, but the ideas generalize to many channels.):

1. What threat scenarios can be mitigated by splitting data among two channels?
2. What data splitting policies are most effective in mitigating these threats?
3. What measures of effectiveness can be employed to compare these policies?
4. What elements are required of a simulation framework to allow for the testing and performance analysis of these policies?

This approach lays the groundwork for future two-channel and multi-channel experimental systems. A proof-of-concept two-channel system was created with a limited number of communication schemes referred to as policies. These policies are used to provide solutions to the three main attack scenarios presented earlier in this chapter: interception, jamming and data injection.

1.5 Expected Contributions

This thesis claims four contributions resulting from the questions presented in the Approach Section 1.4 discussed earlier. These contributions are evaluated using

the two-channel communication system discussed and evaluated in Chapter IV. By answering these questions, this thesis seeks to make the following contributions:

1. Development of a Proof of Concept demonstrating that common threats to data-in-transit can be at least partially mitigated by utilizing two-channel communication methods.

2. At least one data-splitting technique, or policy, that can be utilized to provide increased data-in-transit security for each of the threat scenarios presented.

3. A preliminary framework for measuring the effectiveness of the two-channel policies tested, specifically in regards to data confidentiality, integrity and availability.

4. Proof of concept that these data-splitting policies can be utilized in a dynamic way in which greater data confidentiality, integrity or availability can be provided on demand depending on user requirements.

1.6 Structure

This thesis is organized as follows: Chapter II discusses the preliminary and related works for two-channel communication, data-in-transit security and current solutions. Section III discusses the design and implementation methodology used to construct a simulation of our two-channel communication system to include the policies and packet structure employed. Chapter IV presents an analysis of the design decisions discussed earlier in Chapter III as well as a study of the completed system simulation. Finally, Chapter V concludes the thesis and presents a description of future work.

II. Preliminaries and Related Works

This chapter presents information pertinent to a comprehensive understanding of existing research into two-channel communications theory. It specifically focuses on cyber security research that has explored the added benefits of securing data in transit through the use of multiple communication links.

2.1 Two-Channel Communications

For the purpose of this thesis, a two-channel communication system is defined as any pair of physical or virtual full-duplex communications links. Multiple channels of communication allow for greater diversity in channel characteristics which may ultimately improve the resiliency of the network against common cyber attacks. In the case of a two-channel system, the parallel heterogeneous channels may therefore provide inherently greater security over a single channel, despite the increase in system complexity. [19] For instance, in a wireless system, this diversity in channel characteristics could refer to two parallel links operating simultaneously over different frequencies. A common example would be an 802.11 wireless access point operating on both the 2.4GHz and 5GHz frequency bands. Although employing similar protocols on both bands, the separation in frequency affords greater resiliency to any one type of cyber attack. Additionally, other tactics such as data randomization, deception and adaptive management and response can be employed using two-channel communication to create transformational improvements which increase the adversaries cost and uncertainty, ultimately leading to fewer attacks. [5]

Communications over parallel links is generally implemented for the purpose of redundancy or to improve signal reception and throughput. For example, cell-phone modems and Wi-Fi connected devices take advantage of parallel channels by utilizing

a technology called Multiple Input, Multiple Output (MIMO). Using this technology, multiple physical antennas are used to connect to the cell tower or wireless access point. MIMO can improve throughput and the signal-to-noise ratio. However, in addition to these proven performance gains, there exists the potential for increased data security. This security improvement can be realized by splitting the data across these parallel channels. In this way, it may be possible to thwart various types of attacks against data-in-transit even when other defenses, such as encryption, are not available.[1]

Figure 1 shows a simple two-channel communication system in which the sender, Alice, utilizes a splitter to divide and send data over two links before it is recombined at Bob's end.

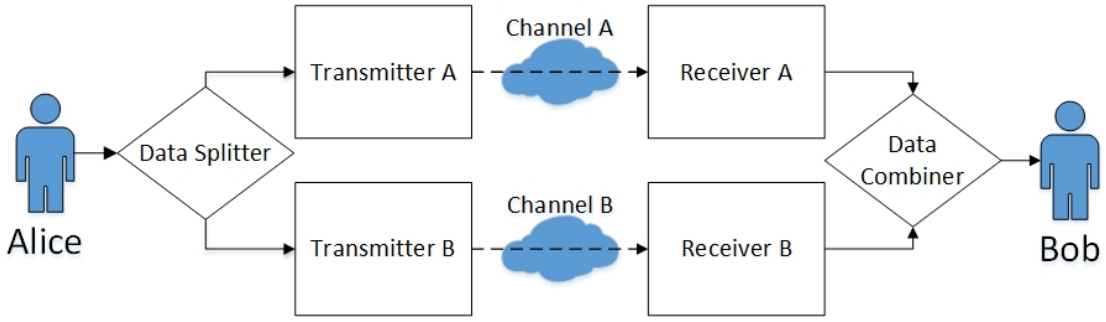

Figure 1. Simple Two-Channel Communication System

2.2 Threats to Data in Transit

This section analyzes the current data-in-transit threat environment. It does not attempt to present all threats to data-in-transit, rather it focuses on presenting only threat information required to fully understand the two-channel solutions presented later in this chapter and the threat scenarios studied later in this thesis.

2.2.1 Interception or Eavesdropping Attack.

Eavesdropping, interchangeably used with interception in this thesis, is the act of passively intercepting private communications between two parties for the purpose of understanding what is being communicated. It violates the confidentiality of the communications. In the classic computer security model, an eavesdropper, referred to as 'Eve', is the party intercepting communications between 'Alice' and 'Bob' as shown in Figure 2. [16]

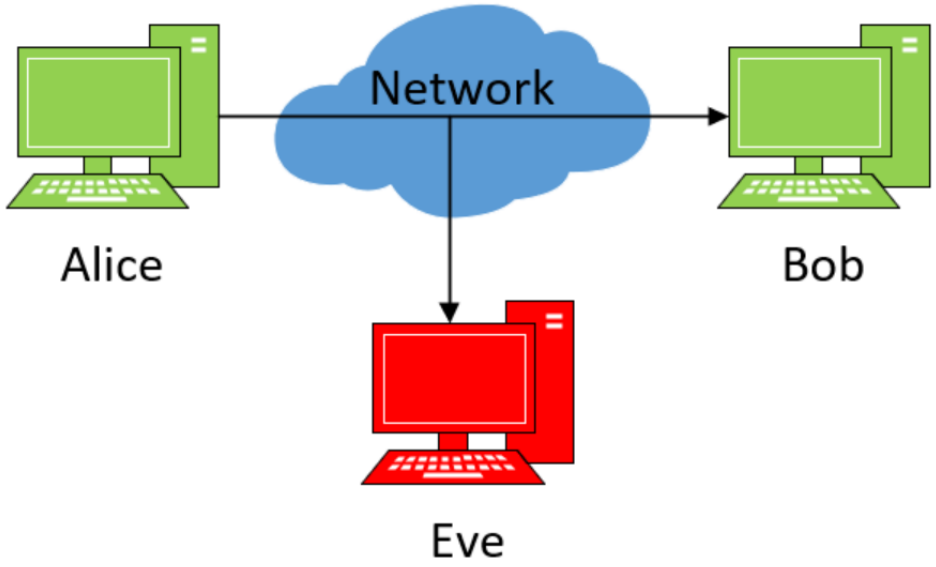

Figure 2. Eavesdropper Attack

In this model, Eve is assumed to have gained access to the communication path between Alice and Bob. In the real-world, Eve could represent another user in-range of a wireless network being used for communication or a user plugged into the same network hub being used to transmit the data in question. For example, an enemy combatant passively intercepting a video feed that is being transmitted in the clear by a military drone, would be considered an eavesdropper.

2.2.2 Jamming Attack.

A jamming event is a type of denial-of-service attack against a wireless medium such as a radio receiver. The goal of this attack is to block communication by impending a target sensors ability to send or receive signals. There are many types of jamming attacks, but this thesis will only discuss the 'constant jammer' model. In this model, the attacker continually emits a signal consisting of random bits which either overpowers or interferes with the legitimate signal at the receiver. When this attack is active, the receiving party cannot decipher the legitimate message being sent by the sender. Figure 3 shows a simple example of a jamming attack targeting the Bob's receiver. [17]

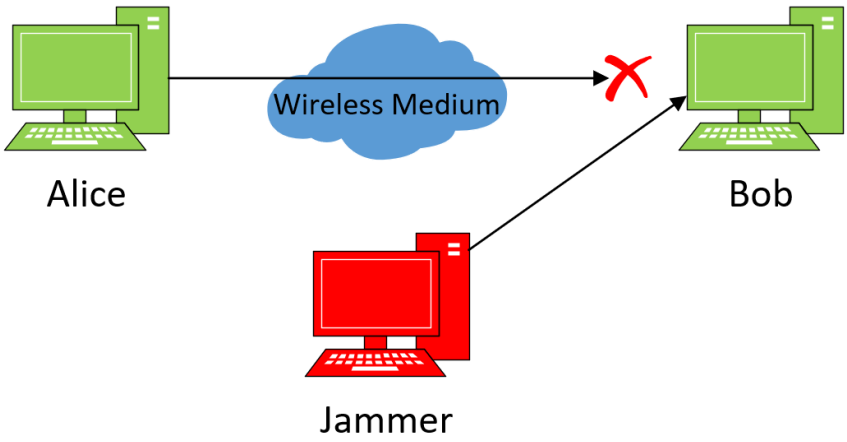

Figure 3. Jamming Attack

2.2.3 Man-in-the-Middle Attack.

The MITM attack (A.K.A monkey-in-the-middle) is similar in nature to the eavesdropper attack presented in Section 2.2.1. However, in this case the attacker is placed between the sender and receiver. Therefore the attacker can not only snoop on traffic, but can also alter traffic before it is seen by the intended recipient. This alteration could include removing a portion of the data, adding data or changing the data in

order to mislead the receiver. Because of the active nature of this attack, the MITM violates both the confidentiality and integrity of the communications. Like the eavesdropper attack, the MITM attack requires the attacker to have access to some portion of the communication path. Figure 4 shows the attacker in-line with Alice and Bob, giving them full access to any communications sent along that path.

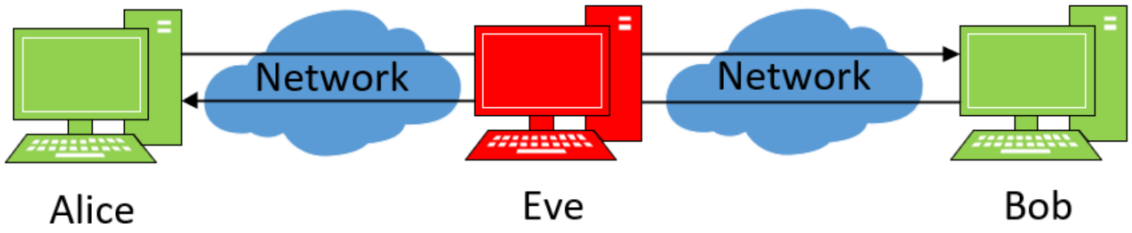

Figure 4. Man-in-the-Middle Attack

It is worth nothing that MITM attacks have become more common with the advent of wireless communications, which are particularly susceptible to eavesdropping because of the broadcast nature of the transmission medium.[3]

2.3 CIA Triad

The Confidentiality, Integrity and Availability (CIA) triad is a model used to guide information security policies. These elements are considered the most crucial components of data security, and therefore information owners must consider them before deciding how best to transmit data. In short, confidentiality involves protecting the contents of the data, integrity ensures the data is unaltered and availability describes the reliability of data access afforded to authorized users. [8]

Additionally, these elements are key metrics to quantifying the overall security posture of a network or communication's link. Often users cannot maximize each of these characteristics without introducing trade-offs that affect the others. Decisions makers must balance the level of each attribute with the mission requirements in

order to achieve the desired result. This thesis will focus on the additional CIA related security opportunities provided to the user as a result of the diversity offered by the addition of a second parallel communications channel. [7]

2.3.1 Confidentiality.

Confidentiality involves protecting data from unauthorized users. It can also be described as information privacy or secrecy. In other words, confidentiality is the measure to which data remains unavailable to eavesdroppers. In order to guarantee confidentiality of data, most modern-day communications utilize encryption. If a sufficiently robust encryption is indeed used, the sender can be confident that an eavesdropper cannot decrypt the data, thus ensuring confidentiality. [8]

In a two-channel system where only one channel is compromised and encryption cannot be used, confidentiality can be obtained to a varying degree by splitting the data between the available channels. The algorithm used to split the data will determine the level of confidentiality that is achieved. In the encryption world, confidentiality can be measured by the type of encryption scheme utilized and the time it would take for an attacker to brute-force the decryption key. However, no corresponding model exists for two-channel systems. This paper will attempt to create a scheme which will enable comparison of the confidentiality associated with various data splitting schemes.

2.3.2 Integrity.

Integrity is the measure of the consistency, accuracy and trustworthiness of the data transmitted over the link. In other words, integrity ensures the data has not been altered between the sender and receiver. To accomplish this, typically an error checking scheme is utilized, such as a checksum or cryptographic hash function. How-

ever, this process is only designed to detect transmission errors or data corruption, as opposed to a sophisticated MITM attack. An attacker could defeat the integrity check by simply updating the checksum to ensure the altered data appeared legitimate. Of course, encryption can help ensure integrity, since the attacker would have to be able to decrypt the data in order to properly modify the contents. [8]

In some scenarios a user may find the encryption overhead unacceptable or may be worried that a MITM attacker could craft fake packets utilizing the receivers public key. In these cases, a two-channel system could also be leveraged to ensure data integrity by utilizing splitting algorithms or by passing the checksum over the second, non-compromised channel. An analog to this is the use of two-factor authentication, though this use case is not explicitly explored in this thesis.

2.3.3 Availability.

Availability is the measure of reliable data access provided to users. In other words, availability ensures that the information can be readily accessed when requested. Typically, availability is more closely associated with network management than network security, however it plays a vital role in the CIA triad since data must be available for it to be of any value. In general, good network management practices are utilized to ensure data availability. This includes utilizing redundant back-ups, ensuring sufficient bandwidth and mitigating denial of service attacks, just to name a few. [8]

With a two-channel system, users can apply the same principles as explained in the MIMO discussion to increase throughput, and therefore availability, by transmitting data over both channels. This type of transmission would theoretically reduce confidentiality and integrity, since an attacker would only have to compromise one channel to obtain all the data. However, by fully utilizing both channels, the sender

would maximize availability.

2.4 Current Solutions

Protecting data in transit from eavesdroppers or MITM attacks is traditionally achieved through the use of encryption. Encryption ensures that only the intended recipient can decrypt the data. This means that even if Eve intercepts the data, she cannot make sense of the data unless she possesses the proper tools for decryption. These typically include knowledge of the encryption scheme and the actual decryption keys.

The Internet makes heavy use of encryption to protect data confidentiality. For instance, most modern websites handling sensitive data utilize the Hyper Text Transfer Protocol Secure (HTTPS). HTTPS in turn uses a robust asymmetric Public Key Infrastructure (PKI) system to ensure only the intended receiver can decrypt the message.

2.4.1 The Cost of Encryption.

Despite the advantages offered by encryption, there are trade-offs in the form of additional overhead costs such as infrastructure requirements, communication latency, bandwidth and energy consumption. For instance, a typical HTTPS web session will require more time and processing power than an equivalent, non-encrypted HTTP session. This is because calculating encryption keys, encrypting data and decrypting data all require additional computational time and power. Likewise, encrypted data takes up more space than its non-encrypted form, requiring more time and resources to transmit the data. All of the trade-offs mentioned become more cumbersome as the robustness of the encryption is increased, which is typically accomplished by increasing the key length. [13] Lastly, encryption does not protect the data from

being intercepted, rather it only attempts to protect the confidentiality of the data by utilizing a proven encryption scheme. For this reason, it is still possible that an eavesdropper could decrypt the data at some point in the future by leveraging advances in computing power or currently unknown vulnerabilities in the encryption scheme.

2.4.2 Encryption in Low-powered Devices.

In applications where processing power and bandwidth are readily available, these overheads may be of little concern. However, in scenarios where these resources are limited, encryption can be a costly endeavor.

For instance, in small low-powered devices, such as those used in networks of remote sensors, the additional computational resources required for robust encryption may consume more time and power than is acceptable.

A sensor network refers to a system of small sensors each containing a general-purpose computing element. Networks such as these are used in numerous industries to monitor conditions or control equipment in remote locations. They often consist of a large number of low-powered devices designed to conserve battery life while communicating critical information over wireless links [9].

Because these wireless sensor networks typically have limited computational and power resources available, modern encryption standards, such as 128-bit advanced encryption standard (AES), can impose a significant burden on the individual nodes. For example, one study found that using 128-bit AES to encrypt one 128-bit block of data takes 1.1 milliseconds (ms), 946 bytes of random access memory and 23.57 microjoules (J) on a IEEE/ZigBee 802.15.4 board commonly used in low-power wireless sensor networks [18]. These resources add up quickly as more data is transmitted over the lifespan of the sensor. Equation 1 shows that after encrypting one gigabyte (GB)

worth of data, the sensor will have expended 0.41 watt-hours (Wh) of energy solely to encrypt the data being sent. This can significantly affect battery life in devices which may only have a few watt-hours of energy.

$$1GB \times \frac{8bits}{1Byte} \times \frac{23.57J}{128bits} \times \frac{1Wh}{3600J} = 0.41Wh \tag{1}$$

Light-weight encryption schemes, such as the SIMON and SPECK encryption ciphers, attempt to solve this issue in low-powered devices by offering more efficient, but less robust encryption options [2]. However, no encryption scheme can reduce the overhead completely. Using two-channel communication, this paper will show that a level of data confidentiality and integrity can be provided without the added overhead of encryption.

2.5 Tunable Security

Tunable security is a security service designed to offer various security configurations that can be selected at run-time. For a security service to be tunable, it must offer at least two security configurations. For instance, when using encryption, a tunable security service might allow a user to select various encryption schemes or key lengths depending on user preference. This would allow the user to tune the service to utilize a more robust encryption when high-security is desired or use a less secure encryption scheme when transmitting data in a low-threat environment. [11]

In the context of this thesis, tunable security will be used to provide the sender multiple methods for securing data in-transit. This will be accomplished by taking advantage of two-channel communication as opposed to more traditional, but computationally expensive methods, such as encryption. Furthermore, the tunable nature of the security service will be achieved by offering multiple options in the form of various algorithms used to determine which bits are transferred over which channels.

In this way, the user can make trade-offs between the elements of the CIA triad in order to match the mission requirements. For instance, in an untrusted environment, the user could tune the system in order to maximize confidentiality, while in a trusted location one could tune the system to ensure better data availability. These tuning actions will be achieved by altering the complexity and behavior of the data splitting algorithm to best match the desired characteristics.

2.6 Cyclic Redundancy Checks

A Cyclic Redundancy Check (CRC) is a common method used in digital networks to detect bit errors in data transmission or data storage. Use of a CRC in data transmission is typically intended only to provide a defense against data corruption. However, as we will show later in this paper, the CRC can also be utilized along with the two-channel system to help detect potential MITM attacks.

For data transmission, error detection is accomplished by first calculating a binary CRC code using the message to be sent and the CRC polynomial specific to whichever CRC algorithm the user chooses. The resulting code is then appended to the end of the message to be sent. Once delivered, the receiver performs the same calculations and compares the result to the appended CRC code. If the results do not match, the receiver knows there was an error in transmission and can request a retransmission. However, if the CRC codes match, the receiver has some level of confidence that the message was not corrupted in transmission.

CRCs vary in length and complexity. Longer CRCs are generally able to detect a higher number of bit errors. The minimum possible number of bit errors that must be injected into the message in order to escape detection by the CRC is referred to as the hamming distance. For example, a CRC with a hamming distance of 4-bits provides error detection for any combination of 1-, 2-, or 3-bit flips resulting from

transmission error. However, there is at least one combination of 4-bit errors that remain undetectable to the CRC. For this thesis, an 8-bit CRC with a hamming distance of 4-bits will be used in conjunction with the two-channel system and is discussed further in Section 3.6.1 [10].

Another technique commonly used in wireless communication is Forward Error Correction (FEC), employed to overcome noise in a transmission medium. FEC may also be employed across multiple channels as a means to detect injection attacks without alerting the MITM attacker. This will be briefly discussed later.

2.7 Chapter Summary

Traditional methods of securing data-in-transit rely on the use of robust encryption. While effective, this technique can be expensive from a computational and bandwidth perspective.

Through the use of a two-channel system, data-in-transit can be secured to varying degrees by splitting the data across both channels. Additionally, the user can tune the level of security by altering the splitting algorithm at run-time in order to match the desired characteristics of the CIA triad.

III. Design and Implementation Methodology

3.1 Objective

The intent of this research is to demonstrate two-channel communication policies that can be employed to improve data-in-transit security. To accomplish this, multiple two-channel communication policies will be developed, simulated and analyzed for effectiveness in various threat scenarios. The results will be used to categorize the advantages and disadvantages of each data splitting policy so that a tunable system can be created to address a fluid threat environment such as those faced by UAVs.

3.2 Assumptions and Constraints

Many of the assumptions arise because this work is conducted in a purely simulated environment which lacks many of the variables and other real-world aspects associated with experimentation. The assumptions and constraints are listed as follows:

- There are exactly two channels capable of equal data rates available. (We are not exploring the concept of passing a key over a low-bandwidth channel). Additionally, these concepts may be applied to n-channels, but we will stick with two channels for simplicity.
- Only one of the two channels may be compromised. The user knows and/or suspects which communication channel may be susceptible to compromise. We will not explore the case in which both channels are compromised, as it would be similar in nature to attack scenarios in which only one channel is available.
- Real-world communication challenges such as channel timing and signal quality are assumed to be non-issues in order to allow a greater focus on the analysis

of the communication policies themselves.

3.3 Design Decisions

There are a number of different approaches which can be utilized to achieve the objectives listed above. One approach is to perform a strictly analytic assessment which would examine the theoretical aspect of the two-channel communication system. While this approach requires the least amount of time and resources, it is the least accurate. Another approach is to simulate the two-channel system using software to emulate all nodes and messages traversing the system. This approach requires more time and resources than the theoretical method, but provides more tangible results in the end. A third approach would consist of actually building out the system in hardware so that the performance could be measured using real-world components operating representative environments. This approach, while the most realistic, is also the most time and resource intensive and least flexible.

This thesis will use the second approach in order to simulate the two-channel system in software. This approach was chosen because it provides the flexibility to quickly simulate and analyze different communication policies while providing more realistic results than a simply theoretical study. Ultimately, the results of these simulations could provide the basis for the construction of a real-world system using actual hardware components operating over different frequencies.

The flowchart shown in Figure 5 below lists the overarching steps taken to untimely achieve a dynamically tunable two-channel communication system. We begin by defining the threat scenarios that will help determine the data-in-transit security requirements. These requirements inform the mitigation techniques that are then refined into data splitting policies designed to take advantage of the two channels available to us. Once the policies have been defined, they are simulated in software

and the results are used to classify their effectiveness and suitability for tuning. Finally, the results are used to build a taxonomy of tunable two-channel communication policies designed to combat the various threat scenarios.

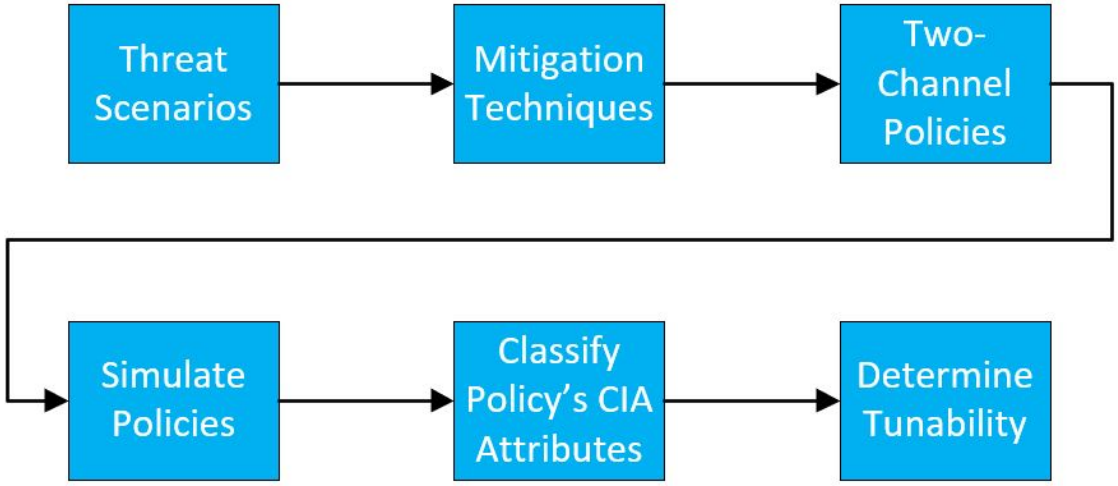

Figure 5. Methodology Flowchart

Of course, the choice to simulate the environment introduces some constraints. Notably, the results of this study will only be as useful as the software used to simulate it. For this reason, the industry standard OMNeT++ software will be used to create the system and provide statistics.

3.4 Scenarios

The following three attack scenarios represent the threat environments applicable to data-in-transit that will be studied in this Thesis.

3.4.1 Eavesdropper.

In this scenario, one channel may be compromised by an eavesdropper. The objective of this scenario will be to maximize confidentiality by minimizing the number

of bits that are intercepted and known by the eavesdropper. The compromised channel may or may not be known.

The flowchart in Figure 6 below shows the adversary's steps during a typical eavesdrop attack on one channel of the communications system.

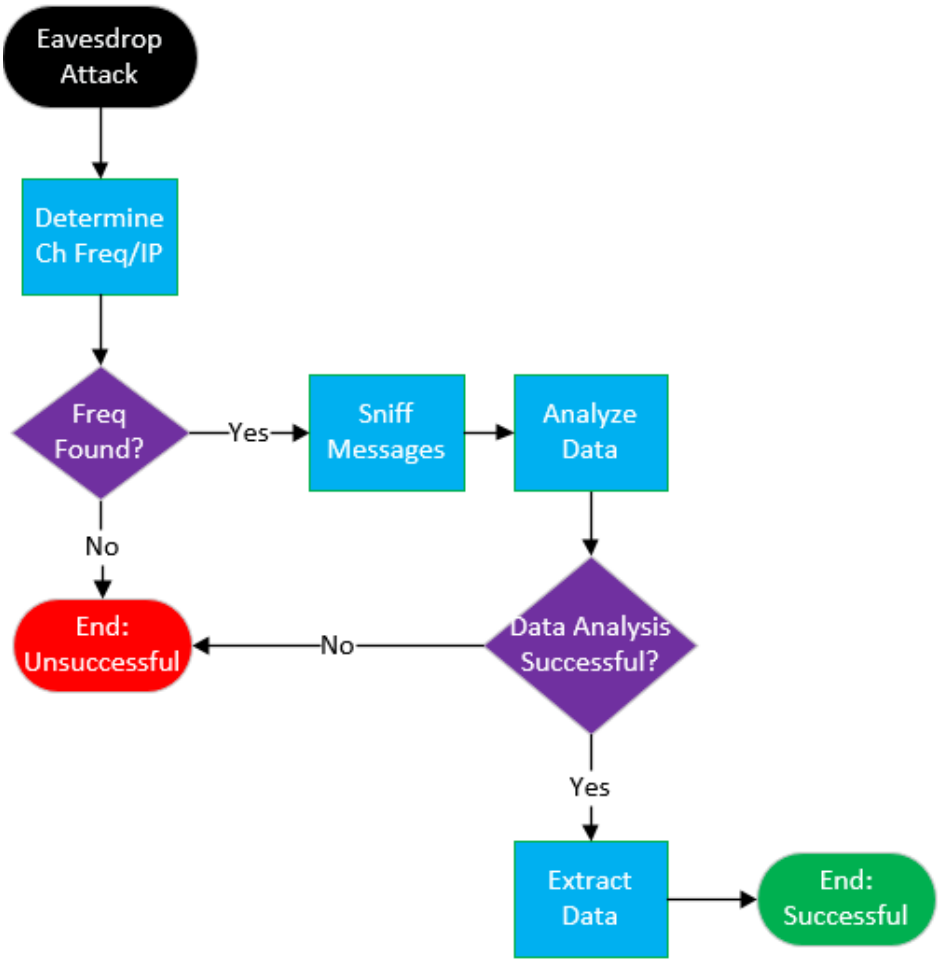

Figure 6. Eavesdropper Attack

This attack methodology was used to develop the three mitigation techniques shown below.

1. Do not send all data on compromised channel.

   Result: Limits the eavesdropper to only partial data, that which is sent over

the compromised channel.

2. Send no data over compromised channel.

   Result: Eavesdropper has no data. But eavesdropper may be suspicious and search for more frequencies if no data is being sent. Data rate is cut in half.

3. Send faux data over compromised channel. Real data over uncompromised channel.

   Result: Eavesdropper has only worthless or misleading data. Data rate is cut in half.

3.4.2 Jamming/Denial of Service.

In this scenario, one channel may become unusable for some period of time due to a jamming or DoS attack. The objective in this scenario is to maximize data availability despite losing the use of one of the channels temporarily or permanently.

The flowchart in Figure 7 below shows the adversary's steps during a typical jamming/DoS attack on one channel of the communications system.

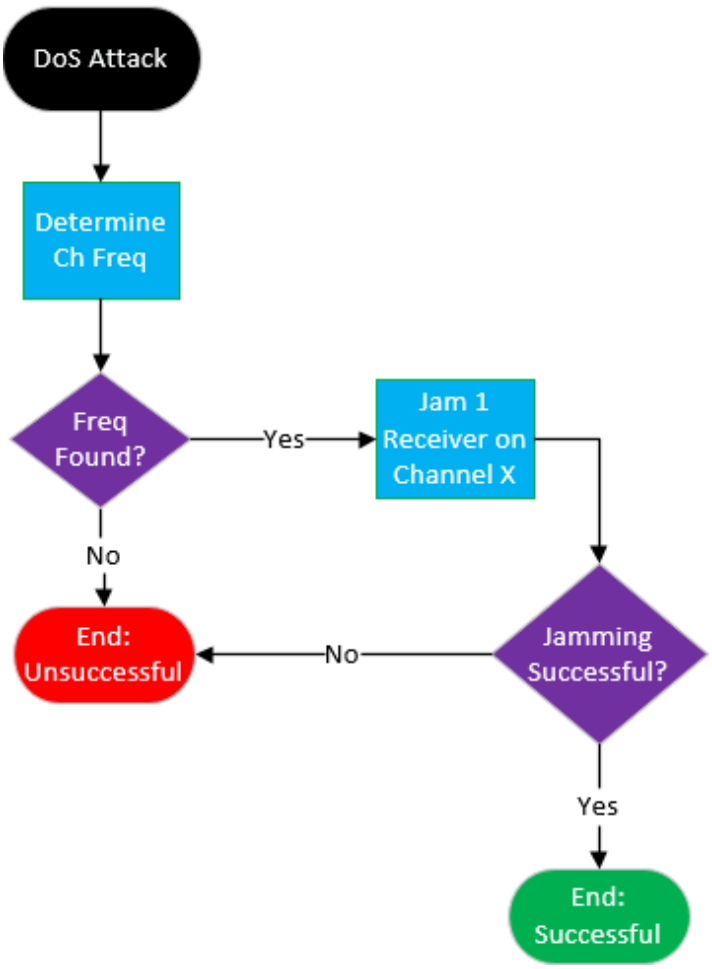

Figure 7. Jamming/Denial of Service Attack

This attack methodology was used to develop the three mitigation techniques shown below.

1. Do not send any data on the jammed channel at any time.

   Result: Jammer may be unaware that any communications are taking place since no activity is observed on the affected channel. Data rate is cut in half since data can only be sent on one channel.

2. Send data over compromised channel until jamming event is detected. Then send data only over the undetected channel.

Result: Jammer sees data on compromised channel before jamming. Executes jamming attack and believes communication to have been denied.

3. Send data over compromised channel until jamming event is detected. Then send data only over the undetected channel. Retry sending data on the jammed channel on a preset schedule to once again take advantage of the compromised channels bandwidth once the jamming event has ended.

   Result: Jammer sees data on compromised channel before jamming. Executes jamming attack and believes communication to have been denied. However, communication resumes at some point after jamming ceases or becomes ineffective.

3.4.3 Man-in-the-Middle.

In this scenario, one of the channels may be susceptible to a man-in-the-middle attack in which the data-in-transit has been modified. The objective is for the system to maximize data integrity by utilizing the two-channels to detect the attack and mitigate the effects.

The flowchart in Figure 8 below shows the adversary's steps during a typical MITM attack on one channel of the communications system.

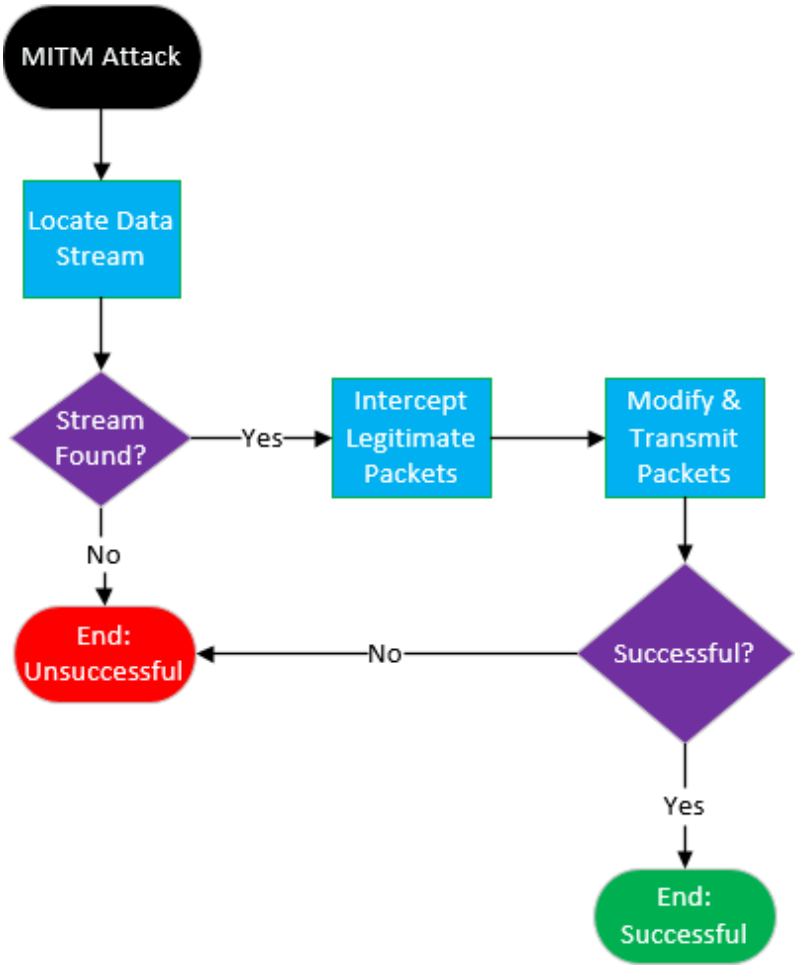

Figure 8. Man-in-the-Middle Attack

This attack methodology was used to develop the two mitigation techniques shown below.

1. Utilize Data CRC in order to detect when data has been modified on one channel. Once attack is detected, cease data transfer over compromised channel. Transfer all data over safe channel.

   Result: MITM will be unable to modify data undetected. However, they may become suspicious if the data flow stops once the MITM attack begins. Additionally, throughput is reduced as only one channel can be used.

2. Utilize Data CRC in order to detect when data has been modified on one channel. Once attack is detected, switch to transferring only fake data over the compromised channel. Transfer real data over the safe channel only.

   Result: MITM will be unable to modify data undetected. Additionally, they may be unaware their attack has been detected if they continue to see data flowing over the compromised channel. However, throughput is reduced as only one channel can be used.

3.5 Policies

For simulation purposes, eight policies were created to correspond to the eight mitigation strategies that were introduced in the previous sub-section. These policies govern the way in which the data bits are divided among the two-channels. Each policy is presented in Table 1 and described in detail below.

Table 1. Data Splitting Policies

Scenario	Policy #	Description
Eavesdropper	0	Split data over both channels.
	1	Send data only over safe channel.
	2	Send real data over safe channel. & fake data over compromised channel.
DoS/Jamming	3	Do not send data on channel susceptible to compromise.
	4	Send data on both channels until jamming is detected, then limit to safe channel.
	5	Send data on both channels until DoS is detected, then limit to safe channel. Periodically retry sending on jammed channel.
MITM	6	Detect data modification w/data-CRC. Once detected, transfer data only over safe channel.
	7	Detect data modification with data-CRC. Once detected, transfer real data only over safe channel & fake data over compromised channel.

The Eavesdropper policies are meant to thwart an attacker who has the ability to observe one channel. The three policies listed provide options to prevent the eavesdropper from obtaining 100% of the data communicated. The first policy ensures only a portion of the data is sent on the compromised channel. The proportion of data sent across this channel can be specified by the user. The type of data may be used to determine the acceptable amount of data compromise that is possible while still maintaining the appropriate degree of confidentiality. The second policy simply

avoids sending data over the possibly compromised channel. While the third policy is similar to the second except that it also provides for sending fake data over the compromised channel. The fake data could provide the eavesdropper with a false sense of success in their attempt to intercept the communications.

The three policies designed to counteract the DoS attack provide options for circumventing the jammed channel. Policy Three simply avoids using the channel susceptible channel all together. Policy 4 operates in a split data mode similar to Policy 0 until a jamming event is detected at which point all data transmission is limited to the safe channel. Policy 5 is similar to Policy 4 except that it intermittently checks the status of the jammed channel and will resume the Policy 0 behavior if it determines the channel is no longer jammed.

The final two policies are designed to detect and thwart MITM attacks. Policy 6 and 7 utilize a cyclic redundancy check which is calculated using 16-bits of data before splitting the data into two 8-bit chunks for transmission across the medium. Modification of the data results in a different CRC being calculated by the assembler which triggers detection of the attack. Policy 6 simply refrains from transferring over the suspect channel after detection of a MITM attack while policy 7 does much of the same with the addition of transferring phony data over the compromised channel for the purpose of deception.

3.6 Packet Structure

For simulation purposes, a rudimentary packet structure was created which could meet the needs of the policies presented in the previous section. The packet structure is shown in Figure 9 and the individual fields are discussed below.

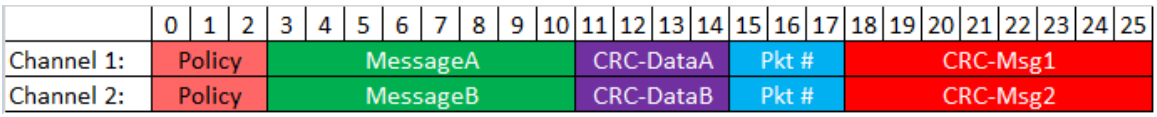

Figure 9. Simulation Packet Structure

1. **Policy** - This 3-bit field is used to specify the policy being used to send the corresponding packet. The policy numbers match those presented in the Section 3.5. (i.e. a binary '101' in this field would represent Policy 5. The assembler uses this field to ensure corresponding packets contain matching policy number before processing them.

2. **MessageX** - This 8-bit field contains the data bits.

3. **CRC-DataX** - This 4-bit field contains the data level CRC for policies requiring detection of MITM attacks (i.e. policies 6 & 7). For all other policies, this field is used to transmit an additional 4-bits of data.

4. **Pkt #** - This 3-bit field is used to record the packet number. Packets being sent over either channel will have a matching packet containing an identical packet number sent over the opposite channel. These packet numbers are used to ensure similar packets are processed together by the assembler according to their policy.

5. **CRC-Msg#** - This 8-bit field is used for error detection during message transmission. The CRC corresponds to the entire packet and is checked for correctness by the assembler upon arrival.

It should be noted that these packets are representative of data being transmitted a the physical layer of the Open Systems Interconnection model. This allows for normal network traffic (i.e. Transmission Control Protocol, Internet Protocol) to ride on top of the two-channel system.

3.6.1 CRC Description.

The CRC chosen for use in the simulation packet design was the 8-bit long CRC-8 polynomial represented by the hex value 0xEA. This polynomial was chosen because it provides a hamming distance of 4-bits for messages up to 85-bits in length while only adding a relatively short 8-bits of information to the packet. Therefore, it provided a good balance between length and performance. This CRC scheme was employed for both the message and data CRC for simulation simplicity although different CRCs could be utilized for each of these fields. The CRC-8 scheme is also commonly used in commercial communication technologies such as the Digital Video Broadcasting satellite second generation standard and is well understood.[10]

3.7 Policy Comparison Framework

3.7.1 CIA Attributes - Common Vulnerability Scoring System.

The policies can be compared to each other based on the three CIA attributes described in Section 2.3.1. The framework used to compare them is a modified subset of the industry standard Common Vulnerability Scoring System (CVSS) v2.0 normally used to assess the severity of computer system security vulnerabilities. As part of assessing vulnerabilities, the CVSS assess CIA attributes as impact metrics and uses a three-tiered scale consisting of Complete (C), Partial (P) and None (N). Where (C) corresponds to a complete loss of an attribute, (P) represents a partial loss of the attribute and (N) refers to no loss. [4]

For the purpose of this thesis, policy comparison between the CIA attributes of the various policies presented will consist of (C) for a complete loss and (N) for no loss. While a partial loss of an attribute will be represented by a numeric value between 1 and 99 which corresponds to the percentage of the resource lost. For

instance, if a policy is found to provide 100% confidentiality in a given scenario, an (N) is used to represent that there is no loss of confidentiality. Likewise, a (C) for integrity represents a complete loss. Finally, a numeric value in either integrity or confidentiality represents the percentage of the message which was compromised. While a numeric availability value represents the percentage of throughput which was lost in relation to the maximum provided by Policy 0.

3.8 Software

1. Windows 10 - An industry standard Operating System to support all other software described below.

2. OMNeT++ v5.2.1 - This was the software used to simulate the two-channel system. OMNeT++ is a discrete event simulator which utilizes a C++ based library built specifically for network simulation. OMNeT++ provides for the ability to create and configure models, perform batch executions and analyze simulation results.

3. OMNeT++ IDE - The OMNeT++ Integrated Development Environment is based on the eclipse platform and was used to code the simulation in C++.

IV. Simulation and Analysis

4.1 Introduction

Using OMNeT++, the seven policies presented in Chapter 3 were simulated in software. These policies were then compared using their measured throughput. Additionally, these policies were examined for potential effectiveness and vulnerabilities.

4.2 Simulation Design

The simulation was programmed using C++ and consisted of six main components. Figure 10 shows each of these components parts connected in a two-channel system used in the simulation environment.

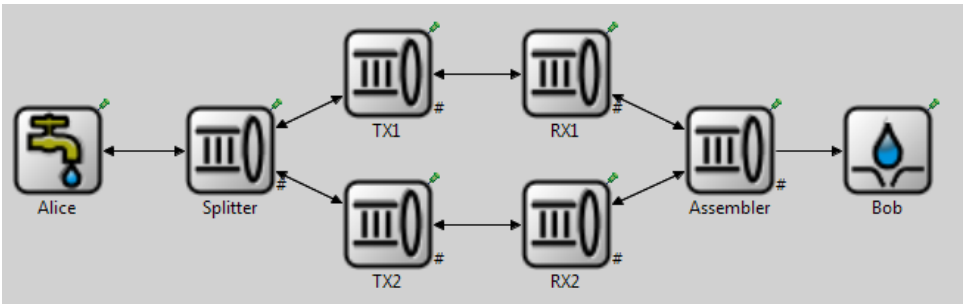

Figure 10. Two-Channel Simulation Structure in OMNeT++

The individual components that comprise the simulation scenario are defined in more detail below.

4.2.1 Component Descriptions.

1. **Alice** - The Alice module represents the sending party and provides the bits to be transmitted. This simulates the message that needs to be communicated to the receiving party, Bob.

2. **Splitter** - The Splitter module is where the data splitting logic is implemented. It is here that decisions on which data to send over channel A or B are made based on the user input provide.

3. **Transmitter "TX# "** - The two transmitter modules represent the physical transmitters that send data over each of the channels.

4. **Receiver "RX# "** - The two receiver modules represent the physical receivers that receive data over each of the channels.

5. **Assembler** - The Assembler module executes logic much in the same manner as the Splitter module, in that it must assemble the message after transmission and according to the policy being used.

6. **Bob** - The Bob module represents the receiving party and ultimately receives the message being transmitted after it is assembled in the Assembler module.

4.2.2 User Input.

The user provides a plain-text message to be transmitted from Alice to Bob. This plain-text message is then converted into American Standard Code for Information Interchange (ASCII) bits by the program and transmitted according to the user specified policy. The user specifies this policy by entering a number between 0 and 7 before program execution. This number corresponds to the policy number shown in Table 1.

4.2.3 Simulation Operation.

The OMNeT++ simulation begins with the Alice block. This block reads in data from the input text file, described in Section 4.2.2, one character at a time. It then

converts the ASCII character to binary and passes the binary string bit-by-bit to the Splitter.

The Splitter reads-in the policy number being simulated from a the text file set by the user. It then buffers data from Alice until enough data is received to transmit. The number of bits required for transmission depends on the policy being simulated.

The Splitter will then load the data into 26-bit message buffers based on the set policy. Once the data bits are set, the Splitter will set the policy bits, packet number bits and calculate the CRC-Msg bits using the CRC-8 scheme discussed in Section 3.6.1. For some policies, the Splitter creates a packet for each channel and sets a matching packet number in each message that will be used by the Assembler to verify the pair of packets correspond to one another. For other policies, where only one channel is in use, the Splitter only creates one packet to send over a single channel.

Once the packets are created, the Splitter passes the messages to the TX blocks to be sent across the simulated channel. The TX block then transmits the message bit by bit across the simulated delay channel to the corresponding RX block. The splitter also stores a copy of the packet(s) in-case a retransmission is requested by the Assembler.

Once the packet is received by the RX block, it is passed ot the Assembler which buffers the bits until an entire packets is received. Once received, the Assembler checks for transmission errors by calculating the packet's CRC-Msg bits and comparing them to the bits received. If the bits do not match, the Assembler sends the Splitter a single one bit to request a re-transmission of the packet. However, if the CRC-Msg bits do match, the Assembler replies with a zero-bit acknowledgment on the corresponding channel. The Splitter will wait to send the next packet(s) or retransmit the last packet until it receives the acknowledgment from the Assembler.

Finally, the Assembler will process the successfully transmitted packet(s), accord-

ing to the policy identified by their set policy field bits, in order to extract the included data. The program then turns the received data bits into their corresponding ASCII letters and prints the final message to the screen.

4.2.4 Simulation Output.

The simulation outputs multiple text files to the directory of choice for later analysis. These text files include a record of data sent over each channel as seen by each of the blocks identified in Figure 10. Additionally, the Splitter and Assembler blocks also output key system messages such as acknowledgments, re-transmission requests and state transitions to the OMNeT++ terminal for use in debugging and tracking behavior. Finally, the assembler also outputs the received message packets along with the associated in ASCII characters and timestamps used for calculating throughput.

4.2.5 Simulated Policy Behaviors.

The following subsections describe the test policies programmed behaviors in more depth.

4.2.5.1 Policy 0.

In Policy 0 the sender transmits data across both channels in a 50/50 split. The first byte of data is stored in the Message A field and the second byte of data is sent in the MessageB field. An additional third byte of data is split evenly across the CRC-DataA and CRC-DataB fields. In the case of Policy 0, the CRC-Data fields are utilized for data transmission rather than storing of the CRC information. This is possible because the data level CRC is not required for any policies except those designed to counteract the MITM scenario. Rather than changing the packet structure on a per

policy basis or letting the CRC-Data field go unused, it is re-purposed for additional data throughput. In this way, the bit-rate of this policy is increased by 50% since each packet can transmit 12-bits of data versus 8-bits otherwise.

4.2.5.2 Policy 1.

Policy 1 follows the same structure as Policy 0, however it limits packet transmission to channel A, which is assumed to be the safe channel in the simulation. Therefore, the first byte is stored in channel A's MessageA field and the first half of the second byte is stored in the CRC-DataA field. NO packets are sent across channel B.

4.2.5.3 Policy 2.

Policy 2 is similar to Policy 1 in that it sends data over the safe channel A. However, random data is also sent by the program over the compromised channel B in order to deceive the eavesdropper. The Assembler simply ignores the data received over channel B when Policy 2 is specified.

4.2.5.4 Policy 3.

Policy 3 was designed to be the safe counter to a possible jamming event on the susceptible channel B. However, it is the same as Policy 1 in practice since it only sends data over the safe channel A.

4.2.5.5 Policy 4.

Policy 4 reacts to a simulated attack by ceasing data transmission over channel B once a jamming event is detected. The simulated jamming event is characterized by the receiver failing to receive packet on three successive attempts. Once the Splitter

receives the third successive retransmission request from the Assembler, it reverts to sending data over the single non-jammed channel. In other words, the system resorts to Policy 3 in order to complete the simulation after a jamming event is detected.

4.2.5.6 Policy 5.

Policy 5 behaves in a similar manner to Policy 4 except that it periodically retries sending on the jammed channel. After a jamming event is detected, the policy reverts to the behavior of Policy 3 as in the above. However, after a preset number of successful transmissions over the safe channel, the system attempts to transmit again over the jammed channel. For simulation purposes, the system will attempt to retransmit on the jammed channel after 10 and 30 successful packets sent over the clear channel. If transmission over the jammed channel proves successful on any of the first three attempts, the system will resort to its original state and continue to the transfer over both channels until another jamming event is detected. Figure 11 shows a simple state diagram to illustrate the programmed behavior of Policy 5. The 'FailCounter' variable is used to track the number of failed transmission attempts while the 'Counter' variable is used to track the number of successful one-channel transmissions.

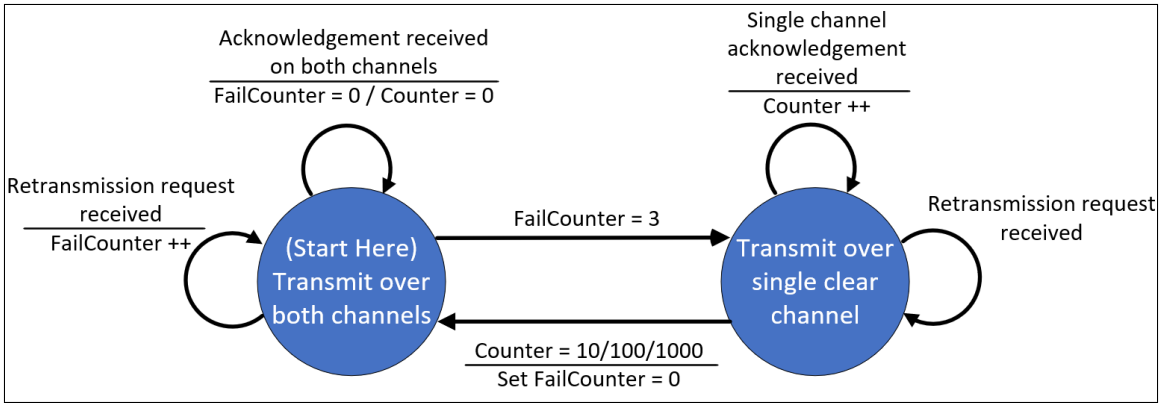

Figure 11. Policy 5 State Diagram

4.2.5.7 Policy 6.

Policy 6 is programmed to send data over both channels like in Policy 0. However, the policy also utilizes the CRC-8 scheme to create the data CRC bits to fill the CRC-DataA and CRC-DataB packet fields. This is done by generating an 8-bit CRC using the 16-bits of data (MessageA + MessageB) as an input. Then placing the first 4-bits of that CRC into the CRC-DataA field and the last 4-bits into the CRC-DataB field. On the receiving end, the assembler performs the same operation using the 16-bits of received data as the input. The assembler then compares the resultant CRC bits to the received CRC bits to determine if there are any differences. Since only one channel is susceptible to manipulation, an attacker could not modify the half of the data CRC sent on the safe channel and therefore any changes to the data should result in a different data CRC. If this occurs, the assembler assumes a MITM attack has occurred and requests retransmission on both channels. If the assembler receives three retransmission requests in a row on both channels, it will resort to sending data only on the safe channel for the remainder of the simulation, similar to Policy 1.

4.2.5.8 Policy 7.

Policy 7 operates in much the same way as Policy 6. However, once the MITM attack has been detected, the system transitions to the behavior of Policy 2 in that random data is also sent over the compromised channel for the remainder of the simulation in order to deceive the attacker.

4.2.6 Simulating Attacks.

For each of the policies listed in Section 4.2.5, there exist a corresponding attack matching the threat scenario shown in Table 1. These attacks are programmed into each policy's OMNeT++ function and can be enabled by changing a switch in the

code. Additionally, attacks such as jamming can be tunned to occur more often or for longer durations. The program will also output status update messages to the terminal when the attacks are taking place and when the policies detect an attack occurring.

4.3 Policy Comparison

4.3.1 Eavesdropper Scenario.

The three policies developed to counter the eavesdropper attack are compared in Table 2 using the subset of the CVSS framework discussed above. Note that the eavesdropping attack targets only the confidentiality of the communications. In Policy 0, there is a partial loss of confidentiality of the transmission because the attacker will intercept all of the data sent across the tapped channel. However, policies 1 and 2 have no impact to confidentiality because the data is sent over only the safe channel. The trade-off for limiting data to the safe channel can be seen in the effect on availability as 50% of the throughput is lost when limiting transmissions to just one channel. Integrity is not affected in this scenario since the adversary is limited to eavesdropping.

Table 2. Impact Scores for Eavesdropper Policies (Critical Info)

Policy	Description	C	I	A
0	Split data over both channels.	50%	N	N
1	Send data only over safe channel.	N	N	50%
2	Send real data over safe channel & fake data over compromised channel.	N	N	50%

The comparison presented in Table 2 can be used to determine the best policy choice given the data protection requirements. For data that is confidential in nature, it may be preferable to utilize Policy 1 or 2. Whereas Policy 0 would be the best choice for less critical data or in cases where throughput is valued over confidentiality.

4.3.2 DoS/Jamming Scenario.

The three policies developed to counter the jamming attack are compared in Table 3. Since the jamming scenario incorporates the eavesdropper and jamming attacks, trade-offs can be seen between both the confidentiality and availability. Integrity remains unchanged since the adversary is not capable of MITM attacks in this scenario. Policy 3 ensures no impact to confidentiality and integrity but sacrifices availability in the form of throughput just as with Policy 1 and 2. Policy 4 and 5 act in different manners depending on the state of the attack. Before the jamming event is detected, the policies operate over both channels and therefore there is no negative impact to throughput, however confidentiality is still impacted since the adversary could potentially eavesdrop on the tapped channel. Once the system detects a jamming event, Policy 4 switches modes to cease transmissions over the jammed channel. This results in an impact to the availability by cutting the throughput in half. Policy 5 initially reacts in the same manner, however it has the potential to resort to the previous state when the jamming event is no longer detected. It attempts to send a packet on the jammed channel at user specified intervals. If successful, the policy reverts back to it's pre-attack state until another attack is detected.

Table 3. Impact Scores for DoS/Jamming Policies

Policy	Description	Pre-Attack			Post-Attack		
		C	I	A	C	I	A
3	Do not send data on channel susceptible to compromise.	N	N	50%	No Change		
4	Send data on both channels until jamming is detected, then limit to safe channel.	50%	N	N	N	N	50%
5	Send data on both channels until jamming is detected, then limit to safe channel. Periodically retry sending on jammed channel.	50%	N	N	N	N	50%

The simulation output also allows us to visualize the policy's throughput in respect to one another. The major difference in the throughput of the jamming policies is found in comparing Policy 4 and 5. Policy 4 reverts to using one channel for the remainder of the simulation once a jamming event has been detected. This reduces the throughput by half. Policy 5 behaves in much the same way except it will attempt to re-send over the jammed channel after a period of time. This gives Policy 5 the potential to re-acquire some of it's lost throughput after the jamming attack has ceased.

Figure 12 shows a comparison of policies 4 and 5 in a jamming scenario in which there are two main jamming events. The plot was obtained by graphing the OMNeT++ simulation output timestamps using Microsoft Excel©. It compares the number of bytes successfully transmitted and their timestamps to show which pol-

icy is achieving a greater throughput. As can be seen from the graph, both policies react to the initial jamming even, represented by the two vertically shaded areas, by switching to one-channel transmission and thus reducing their throughput. However, Policy 5 is programmed to re-attempt transmission on the jammed channel after transmitting 10 packets using only the safe channel and again after 30 packets. Since the first jamming event is relatively short, the initial attempt to re-utilize the previously jammed channel proves successful and Policy 5 begins to re-use second channel, increasing it's throughput. Another jamming event strikes at around 100 seconds of simulation time and lasts for just over 30 seconds. In this case, the policy again attempts to re-transmit over the jammed channel after 10 packets and finds that it is still jammed. It then reverts to sending over only the safe channel for another 20 packets before ultimately attempting to use channel B again. On the second attempt, it is successful and begins transmitting at the higher rater once more.

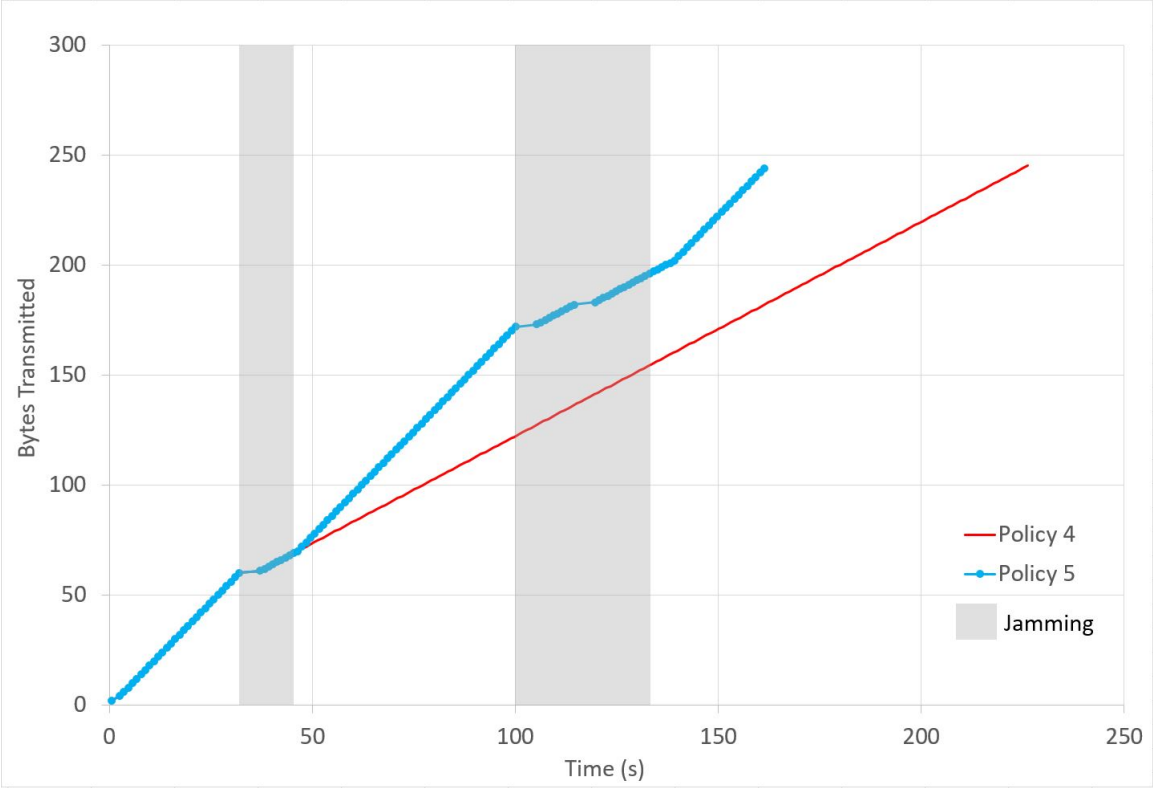

Figure 12. Jamming Scenario Throughput Comparison

4.3.3 MITM Scenario.

The two policies developed to counter the MITM scenario are compared in Table 4. Additionally, Policy 0 is included to illustrate differences in the integrity measure. Note that Policy 0 provides no protection against modification of the data in-transit by the attacker. Instead, it provides superior throughput since the data-CRC field is used to transmit data instead of a dedicated CRC. Policy 6 and 7 provide protection against data modification via the data-CRC while sacrificing throughput both before and after an attack is detected.

Table 4. Impact Scores for MITM Policies

Policy	Description	Pre-Attack			Post-Attack		
		C	I	A	C	I	A
0	Split data over both channels	50%	P	N	No Change		
6	Send data over both channels until data-CRC detects data modification. Then transfer only over safe channel.	50%	N	33%	N	N	67%
7	Send data over both channels until data-CRC detects data modification. Then transfer only over safe channel & fake data over compromised channel.	50%	N	33%	N	N	67%

4.4 Policy Selection

Table 5 shows all eight policies listed along with their rated performance in each of the attack scenarios. The green highlighting represents the best performance possible for the policy against that scenario while the red squares represent a policy which suffers a complete impact from the scenario. Those cells that show two ratings separated by a slash represent the rating of the policy before and after the attack. Finally, the blue highlighted cells represent a policy which performs best, but only either before or after the attack.

By examining this table, it becomes possible for the user to select a policy that best first their data requirements and potential threats. For instance, we can see that Policy 0 would not be a good choice in a possible jamming scenario since it suffers a complete loss of availability after a jamming attack. This is because the policy

is unable to detect a jamming attack and instead will attempt to retransmit on the affected channel. In another example, we can see that if the threat scenario includes a potential MITM attack and the user wishes to ensure integrity while maximizing throughput to the greatest extent possible, then Policy 6 and 7 are the best options. This is because the policy will suffer less impact to throughput at only 33% after an attack has begun, compared to 50% for other high-integrity policies such as Policy 1 and 2 which utilize only the safe channel.

Table 5. All Policy Performance Compared

#	Eavesdropper				Jamming				MITM			
	C	**I**	**A**	**D**	**C**	**I**	**A**	**D**	**C**	**I**	**A**	**D**
0	50%	N	N	No	50%	N	N/C	No	50%	50%	N	No
1	N	N	50%	No	N	N	N	No	N	N	50%	No
2	N	N	50%	Yes	N	N	50%	Yes	N	N	50%	Yes
3	N	N	50%	No	N	N	50%	No	N	N	50%	No
4	50%	N	N	No	50%/N	N	N/50%	No	50%	50%	N	No
5	50%	N	N	No	50%/N	N	N/50%	No	50%	50%	N	No
6	50%	N	33%	No	50%	N	33%/C	No	50%/N	50%/N	33%/67%	No
7	50%	N	33%	No	50%	N	33%/C	No	50%/N	50%/N	33%/67%	Yes

V. Conclusion and Future Works

5.1 Conclusion

This proposed two-channel communication system may potentially benefit DoD operations which utilize communication in contested environments. For UAVs operating over hostile territory, this could mean implementing more secure communications at times when encryption is not yet viable or resource effective enough to justify it's use.

Our contribution of a policy comparison framework provides users a starting point in which to classify and compare policies for use in policy selection given the data security requirements and potential attack scenarios. The framework takes into account the main tenants of data-in-transit security while allowing users to expand on it to incorporate other attributes if desired.

Additionally, the OMNeT++ simulation space provides future researchers a flexible and extensible way in which to develop and test their policies in a variety of scenarios. The system allows for testing of attacks against policies and evaluation fo the results by examining data across either channel or performance characteristics such as throughput.

In future work, the goal will be to improve on the range of policies available to combat attack scenarios while exploring other threat models that may benefit from the added security of two-channel data splitting. Lastly, moving the two-channel system out of simulation and into live testing using hardware is a logical next step.

5.1.1 Research Questions.

The eight proof-of-concept policies along with the policy comparison framework and simulation environment provide a means to answer the research questions poised

in Section 1.4.

Based on evaluation of the policies discussed in Section 4.3, the first research question can be answered in part, although future work may provide additional answers to this question. This research showed that, at a minimum, the three studied threat scenarios to include eavesdropping, jamming and MITM attacks can be mitigated to varying degrees by implementing the two-channel data-splitting policies developed.

The second and third questions examine the data-splitting policies themselves and can be answered using the policy comparison framework presented in Section 3.7. The framework provides a standardized method in which to rate each security attribute of the policy. As shown in Section 4.4, the best policy for each combination of data requirements and threat scenarios can be identified after running the policies through the developed comparison framework.

The last question can be answered using the OMNeT++ simulation space developed for this very purpose. The software-based policy and threat model implementations provide a place to build and test policies in a configurable two-channel system without the need for more time consuming and expensive hardware components.

5.1.2 Contributions.

The contributions of this thesis are threefold: Create proof-of-concept data splitting policies that can be used to improve data-in-transit security, develop a framework for comparing these policy's CIA attributes and construct a simulation in which the policies can be implemented and evaluated.

The eight policies presented were designed to thwart common cyber security attack scenarios that exist in the DoD threatscape today. These policies, while simplistic, provide a starting point for future research on the subject.

The policy comparison framework proposed can be utilized by future researchers

to compare and contrast the CIA attributes and tunability of more complex policies for the threat scenarios discussed herein or additional scenarios that were not studied in this thesis.

Finally, the OMNeT++ simulation environment provides a basic proving ground in which to construct and run future policies against these same scenarios. The program is not all encompassing in its current form, but it can be expanded on to provide additional data analysis and serves as a starting point for future research in the field.

5.2 Future Work

Future research into this topic area could continue down a number of different avenues. First, this thesis presented three different overarching attack scenarios for which two-channel policies were developed to counteract. However, there are a number of additional attack methodologies that could potentially be mitigated through the use of multi-channel communication schemes. Numerous different threats to communications over wireless networks are presented in [12] and serve as a good starting point to the study of additional scenarios.

In addition to more scenarios, more policies beyond the eight presented in this thesis could be developed ot counteract the scenarios presented. The characteristics of these policies could then be compared against the existing policies to determine if they offer any advantages. Moreover, the simulation could be improved upon to automatically select a policy based on the expected, or even detected, threat environment.

Another fascinating study would include an analysis of data splitting schemes among various file types. The results of this research would allow for the creation of smarter data splitting policies that could be deployed based not only on the scenario

and data requirements, but also on the type of data being transmitted. For instance, if a sensitive MPEG video feed was being transmitted over-the-air, this type of research would ensure the video data is split among the available channels in a way that ensures maximum confidentiality.

Beyond the above, one could study the energy overhead required for utilizing a second channel and compare it to the energy saved by not forgoing robust encryption schemes. This could be accomplished using simulation software or even by building out the system in hardware.

Appendix A. Environment Setup

The software development and policy simulation conducted in Section IV was completed on a 64-bit Windows 7 Service Pack 1 virtual machine (VM) running on the freely available Oracle VirtualBox® software. The environment was configured to allow the VM the use of eight central processing unit cores, 6 GB of RAM and 60 GB of hard drive space. This appendix lists the general instructions required ot setup and configure the OMNeT++ v5.2.1 simulation environment once the operating system has been installed.

1. Download OMNeT++ Discrete Event Simulator from https://www.omnetpp.org/omnetpp
2. Following the Windows instructions in Chapter 2 of the OMNeT++ installation guide found at https://omnetpp.org/doc/omnetpp/InstallGuide.pdf. OMNeT++ can also be installed on other operating systems using this guide.
3. Once installed, start OMNeT++ by double clicking the 'omnetpp.exe' file or running it from the command line. This executable should be located at C:\omnetpp-5.2.1\ide\omnetpp.exe assuming OMNeT++ was installed on the C: drive root directory.
4. The first time the IDE is opened, it will prompt the user to choose a workspace directory. The 'samples' workspace can be chosen by default or the user may create their own directory to be used as a workspace.

Once OMNeT++ is installed and the IDE is open, the policy simulation project can be imported and started with the following commands.

1. Click 'File' in the top left corner of the IDE
2. Select 'Import'

3. From the pop-up box, select 'General' and highlight 'Existing Projects into Workspace', then press 'Next'

4. On the next screen, select the radio button for 'Select archive file:'

5. Click 'Browse' and use the pop-up box to find and select the archive file provided with this thesis (policy_comparison.zip)

6. Select 'Finish'

OMNeT++ should now import the archived project and display it in the Project Explorer window in the IDE. To run the program, right click the project and select 'Run As' and then 'OMNeT++ Simulation'.

Bibliography

1. Marco Baldi, Marco Bianchi, Franco Chiaraluce, Nicola Laurenti, Stefano Tomasin, and Francesco Renna. Secrecy transmission on block fading channels: Theoretical limits and performance of practical codes. *arXiv preprint arXiv:1305.3055*, 2013.

2. Ray Beaulieu, Stefan Treatman-Clark, Douglas Shors, Bryan Weeks, Jason Smith, and Louis Wingers. The simon and speck lightweight block ciphers. In *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*, pages 1–6. IEEE, 2015.

3. Matthieu Bloch, João Barros, Miguel RD Rodrigues, and Steven W McLaughlin. Wireless information-theoretic security. *IEEE Transactions on Information Theory*, 54(6):2515–2534, 2008.

4. FIRST.Org, Inc. *Common Vulnerability Scoring System 3.0: Specification Document*, 6 2007.

5. Harriet Goldman, Rosalie McQuaid, and Jeffrey Picciotto. Cyber resilience for mission assurance. In *Technologies for Homeland Security (HST), 2011 IEEE International Conference on*, pages 236–241. IEEE, 2011.

6. Siobhan Gorman, Yochi J Dreazen, and August Cole. Insurgents hack us drones. *Wall Street Journal*, 17, 2009.

7. Jeff Hughes and George Cybenko. Quantitative metrics and risk assessment: The three tenets model of cybersecurity. *Technology Innovation Management Review*, 3(8):15, 2013.

8. INFOSEC Institute. *The CIA Triad*, 2017 (accessed October 1, 2017).

9. Hemanta Kumar Kalita and Avijit Kar. Wireless sensor network security analysis. *International Journal of Next-Generation Networks (IJNGN)*, 1(1):1–10, 2009.

10. Philip Koopman and Tridib Chakravarty. Cyclic redundancy code (crc) polynomial selection for embedded networks. In *Dependable Systems and Networks, 2004 International Conference on*, pages 145–154. IEEE, 2004.

11. Reine Lundin. *Towards Measurable and Tunable Security*. PhD thesis, Fakulteten för ekonomi, kommunikation och IT, 2007.

12. Teodor-Grigore Lupu, I Rudas, M Demiralp, and N Mastorakis. Main types of attacks in wireless sensor networks. In *WSEAS international conference. proceedings. recent advances in computer engineering*, number 9. WSEAS, 2009.

13. David Naylor, Alessandro Finamore, Ilias Leontiadis, Yan Grunenberger, Marco Mellia, Maurizio Munafò, Konstantina Papagiannaki, and Peter Steenkiste. The cost of the s in https. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, pages 133–140. ACM, 2014.

14. Tingyuan Nie, Lijian Zhou, and Zhe-Ming Lu. Power evaluation methods for data encryption algorithms. *IET software*, 8(1):12–18, 2014.

15. Leon Panetta. Sustaining us global leadership: priorities for 21st century defense. *Washington, DC: US Department of Defense*, 2012.

16. Ed Skoudis and Tom Liston. *Counter Hack Reloaded, Second Edition: A Step-by-step Guide to Computer Attacks and Effective Defenses*. Prentice Hall Press, Upper Saddle River, NJ, USA, second edition, 2005.

17. Wenyuan Xu, Ke Ma, Wade Trappe, and Yanyong Zhang. Jamming sensor networks: attack and defense strategies. *IEEE network*, 20(3):41–47, 2006.

18. Fan Zhang, Reiner Dojen, and Tom Coffey. Comparative performance and energy consumption analysis of different aes implementations on a wireless sensor network node. *International Journal of Sensor Networks*, 10(4):192–201, 2011.

19. Yongguang Zhang, Harrick Vin, Lorenzo Alvisi, Wenke Lee, and Son K Dao. Heterogeneous networking: a new survivability paradigm. In *Proceedings of the 2001 workshop on New security paradigms*, pages 33–39. ACM, 2001.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
22–03–2018	Master's Thesis	July 2016 — March 2018

4. TITLE AND SUBTITLE

Securing Data in Transit Using Two Channel Communication

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Wolfe, Clark L., Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-8865

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-18-M-069

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Intentionally Left Blank

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

Securing data in transit is critically important to the Department of Defense in todays contested environments. While encryption is often the preferred method to provide security, there exist applications for which encryption is too resource intensive, not cost-effective or simply not available. In this thesis, a two-channel communication system is proposed in which the message being sent can be intelligently and dynamically split over two or more channels to provide a measure of data security either when encryption is not available, or perhaps in addition to encryption. This data spiting technique employs multiple wireless channels operating at the physical layer, allowing traditional layers above to run seamlessly over it.
Eight data splitting policies are developed with preliminary evaluation of their effectiveness in combating three common cyber security threat scenarios to include eavesdropping, jamming and man-in-the-middle attacks.

15. SUBJECT TERMS

Data Security, Two-Channel Communication

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Scott Graham, AFIT/ENG
U	U	U	U	62	19b. TELEPHONE NUMBER *(include area code)* (937) 255-6565, ext 4581; scott.graham@afit.edu